Utmpname

Vulnerable to TOCTOU issues

Sean Barnum, Cigital, Inc. [vita[1]]

2007-04-23

Part "Original Cigital Coding Rule in XML"

Mime-type: text/xml, size: 8013 bytes

Attack Category	• Path spoofing or confusion problem
Vulnerability Category	• Indeterminate File/Path • TOCTOU - Time of Check, Time of Use
Software Context	• File Management
Location	
Description	The utmp file keeps track of users who are currently logged in and from where. It has a specific format which contains many utmp structures. The utmpx file is an extension of the original utmp file with it's own extended format. utmpname() and utmpxname() specify the location of each of the respective files. "utmpname()" will be used to refer both functions throughout this document. Any setuid program that runs this as root or as any user with authorization to modify the location of this file must take special precaution.
APIs	**Function Name** / **Comments** utmpname() utmpxname()
Method of Attack	The key issue with respect to TOCTOU vulnerabilities is that programs make assumptions about atomicity of actions. It is assumed that checking the state or identity of a targeted resource followed by an action on that resource is all one action. In reality, there is a period of time between the check and the use that allows either an attacker to intentionally or another interleaved process or thread to unintentionally change the state of the targeted resource and yield unexpected and undesired results. If an attacker can specify the filename used in utmpname() , he or she could append (and create if non-existent) any file with the permissions of the setuid user. An attacker could also have the utmp

1. http://buildsecurityin.us-cert.gov/bsi-rules/35-BSI.html (Barnum, Sean)

data destroyed by specifying that the file should be written to /dev/null.

If a program uses a hard-coded absolute file-path or a relative file-path to a directory that an attacker controls, he or she could create a symbolic link with that same path to a file they wish to append. Thus they could append, say /etc/passwd with utmp-format data.

Finally, since this program is being run with elevated privleges, an attacker could leverage other insecurities in the program.

Exception Criteria

utmpname() can be used safely if the attacker cannot specify or affect the filename and they don't have control over any absolute file-path that may be specified in the program.

Solutions

Solution Applicability	Solution Description	Solution Efficacy
This solution is applicable if you can get away without using utmpname()	Don't use utmpname(). Most systems are fine without altering the location of this file.	100% effective.
This solution is applicable if symbolic links, file-paths to insecure directorys, and user-specified locations are unnecessary.	Don't use symbolic links or file-paths to a directory that an attacker controls or could control. Also, don't use user input to determine the location of the files.	100% effective.
This solution is applicable if portions of the program can be executed with lower privileges.	Drop to a less-privileged user when elevated privileges are not necessary.	This will improve the overall security of a program but will not directly enhance the security of the utmpname() function.
Generally applicable.	The most basic advice for TOCTOU vulnerabilities	Does not resolve the underlying vulnerability

		is to not perform a check before the use. This does not resolve the underlying issue of the execution of a function on a resource whose state and identity cannot be assured, but it does help to limit the false sense of security given by the check.	but limits the false sense of security given by the check.
	Generally applicable.	Limit the interleaving of operations on files from multiple processes.	Does not eliminate the underlying vulnerability but can help make it more difficult to exploit.
	Generally applicable.	Limit the spread of time (cycles) between the check and use of a resource.	Does not eliminate the underlying vulnerability but can help make it more difficult to exploit.
	Generally applicable.	Recheck the resource after the use call to verify that the action was taken appropriately.	Effective in some cases.
Signature Details	void utmpname(const char *file); void utmpxname(const char *file);		
Examples of Incorrect Code	`/* Example of using utmpname() with an argument passed from the command line` `* This argument could be a symbolic link, a relative file path, or an absolute file path, any of which an attacker could control`		

	/ utmpname(argv[1]); / An example of using utmpname() with a relative file path */ utmpname("utmp.new");
Examples of Corrected Code	/* Proper use of the utmpname() command: * We will illustrate dropping privileges until we need our super-user privileges and the proper use of the utmpname() command.*/ uid_t init_uid = geteuid(); // Get the effective user of the running process. This will be the program's user or group owner if setuid or setgid is used. seteuid(getuid()); //Drop to the privileges of the user who is runnig the process. //Do unprivileged tasks... seteuid(init_uid); //Jump back up to a privileged effective user if (utmpname("/var/log/utmp.new") == NULL) return -1; //Return -1 on error seteuid(getuid()); //Drop to the privileges back to those of the user who is runnig the process. //Do more unprivileged tasks.
Source References	• utmpname() man page[2] • utmp man page[3] • utmpx man page[4]
Recommended Resources	
Discriminant Set	**Operating System**: • UNIX (All) **Languages**: • C • C++

Cigital, Inc. Copyright

The Build Security In (BSI) portal is sponsored by the U.S. Department of Homeland Security (DHS), National Cyber Security Division. The Software Engineering Institute (SEI) develops and operates BSI. DHS funding supports the publishing of all site content.

1. mailto:copyright@cigital.com